Raspberry Pi Raspbian tuning / optimising / optimizing for reduced memory usage (dropbear + getty + no ipv6 + dash + swap + noop + overclock + syslogd)

admin

22 July, 2012

Guides are frequerntly updated, please check back

Updated: 2 May 2014

Assumptions

RaspberryPi running Raspbian
Completed and optional server preparation.

Optional: Enable ZRAM (RAM Compression)

Raspberry Pi Raspbian zram compression (compressed swap residing in ram / over allocating memory) completed

Login as root

All the items will need to be done as the root user. Make sure you become root after every restart.

Shell

```
sudo su
```

```
root@raspberrypi:~# free -htl
total used free shared buffers cached
Mem: 215M 46M 168M 0B 8.4M 21M
Low: 215M 46M 168M
High: 0B 0B 0B
-/+ buffers/cache: 16M 199M
Swap: 99M 0B 99M
Total: 315M 46M 268M
```

Replacing OpenSSH with Dropbear | Save: +10MB RAM

This will reduce the memory consumption by 10MB, however you will be sacrificing: ssh portforwarding and ssh logging.
install dropbear
openssh-client is installed to provide SCP support for the dropbear server

Shell

```
apt-get install dropbear openssh-client
```

Stop OpenSSH server
you will not loose your current SSH connection, if you are connected via ssh

Shell

```
/etc/init.d/ssh stop
```

Enable dropbear to start at boot-time

Shell

```
sed -i 's/NO_START=1/NO_START=0/g' /etc/default/dropbear
```

optional: prevent root logins

Shell

```
sed -i 's/DROPBEAR_EXTRA_ARGS=/DROPBEAR_EXTRA_ARGS="-w"/g' /etc/default/dropbear
```

optional: prevent password logins

Shell

```
sed -i 's/DROPBEAR_EXTRA_ARGS=/DROPBEAR_EXTRA_ARGS="-s"/g' /etc/default/dropbear
```

optional: prevent password logins for root

Shell

```
sed -i 's/DROPBEAR_EXTRA_ARGS=/DROPBEAR_EXTRA_ARGS="-g"/g' /etc/default/dropbear
```

optional: prevent root logins and prevent password logins

Shell

```
sed -i 's/DROPBEAR_EXTRA_ARGS=/DROPBEAR_EXTRA_ARGS="-w -s"/g' /etc/default/dropbear
```

optional: change the listening port from 22 to 2222

Shell

```
sed -i 's/DROPBEAR_PORT=22/DROPBEAR_PORT=2222/g' /etc/default/dropbear
```

Start Dropbear

Shell

```
/etc/init.d/dropbear start
```

Test
Make sure you can connect to the server over ssh (using an ssh client).

Disable OpenSSH, but allow it to be used for SSH / SFTP

Shell

```
update-rc.d ssh disable
```

Remove the extra tty / getty’s | Save: +3.5 MB RAM

tty2-tty6 will be disabled. We are keeping tty1 for console, unless you choose to disable it.

Shell

```
sed -i '/[2-6]:23:respawn:\/sbin\/getty 38400 tty[2-6]/s%^%#%g' /etc/inittab
```

optional: disable getty on the Raspberry Pi serial line

Shell

```
sed -i '/T0:23:respawn:\/sbin\/getty -L ttyAMA0 115200 vt100/s%^%#%g'  /etc/inittab
```

Replace Bash shell with Dash shell | Save: +1 MB RAM

Replacing Bash with Dash will increase the system’s overall performance ie. speed up the system boot, reduce disk space, use fewer libraries (save memory) and is more reliable:
Dash is an acronym for Debian Almquist shell (dash). It is a Unix and Linux shell which is much smaller than bash but still aiming at POSIX-compliancy. dash is a POSIX-compliant implementation of /bin/sh that aims to be as small as possible. dash is a direct descendant of the NetBSD version of ash (the Almquist SHell), ported to Linux in early 1997. It was renamed to dash in 2002.

Shell

```
dpkg-reconfigure dash
```

> *Use dash as the default system shell (/bin/sh)? Yes*

Enable a 512MB swapfile

Customize the size of the swap file, 512MB recommended.

Shell

```
echo "CONF_SWAPSIZE=512" > /etc/dphys-swapfile
```

Initialize the swapfile

Shell

```
dphys-swapfile setup
```

Start/enable the swapfile

Shell

```
dphys-swapfile swapon
```

Enable better usage of the swap

Default swappiness is 1, we will change this value to 10, which will allow for better memory usage at the expense of more swap usage. note: this could reduce the life of your sdcard.

Shell

```
sed -i 's/vm.swappiness=1/vm.swappiness=10/g'  /etc/sysctl.conf
```

Purge cached block devices before cached filesystem entries

Shell

```
echo 'vm.vfs_cache_pressure=50' >> /etc/sysctl.conf
```

Optional: Enable Preload to speed up load times | only recommended for desktop usage

Note: this will increase memory usage, **recommended for desktop** to increase overall responsiveness.
Do NOT enable for servers.

Shell

```
apt-get install -y preload
```

Shell

```
sed -i 's/sortstrategy = 3/sortstrategy = 0/g'  /etc/preload.conf
```

Reboot

Shell

```
shutdown -r now
```

Optimize / mount

Shell

```
sed -i 's/defaults,noatime/defaults,noatime,nodiratime/g' /etc/fstab
```

Disable IPv6

Most users will not need ipv6 support and if you are only using a local there is no need. Disable it to save the resources and speed up networking.

Shell

```
echo "net.ipv6.conf.all.disable_ipv6=1" > /etc/sysctl.d/disableipv6.conf
```

Disable the kernel module

Shell

```
echo 'blacklist ipv6' >> /etc/modprobe.d/blacklist
```

Remove IPv6 hosts

Shell

```
sed -i '/::/s%^%#%g' /etc/hosts
```

Reboot

Shell

```
shutdown -r now
```

Overclock cpu, sdram and gpu core without increasing voltage

Overclocking is now Officially Supported.

> *“None” “700MHz ARM, 250MHz core, 400MHz SDRAM, 0 overvolt”*
> *“Modest” “800MHz ARM, 300MHz core, 400MHz SDRAM, 0 overvolt”*
> *“Medium” “900MHz ARM, 333MHz core, 450MHz SDRAM, 2 overvolt”*
> *“High” “950MHz ARM, 450MHz core, 450MHz SDRAM, 6 overvolt”*
> *“Turbo” “1000MHz ARM, 500MHz core, 500MHz SDRAM, 6 overvolt”*

Select one of the following options:

800Mhz permanently: The settings below are safe for all Raspberry Pi’s

Shell

```
echo -e "arm_freq=850\nsdram_freq=450\ncore_freq=350\nforce_turbo=1" >> /boot/config.txt
```

OR

850Mhz permanently: The settings below are only to be used if you have heatsinks installed

Shell

```
echo -e "arm_freq=950\nsdram_freq=500\ncore_freq=450\nforce_turbo=1" >> /boot/config.txt
```

OR

700Mhz-1000Mhz dynamic: Scales the cpu frequency according to the load

Shell

```
echo -e "force_turbo=0" >> /boot/config.txt
```

Reboot

Shell

```
shutdown -r now
```

Replace Deadline Scheduler with NOOP Scheduler

NOOP scheduler is best used with solid state devices such as flash memory.

Shell

```
sed -i 's/deadline/noop/g' /boot/cmdline.txt
```

Reboot

Shell

```
shutdown -r now
```

Replace rsyslogd with inetutils-syslogd and remove useless logs

Reduce memory and cpu usage. We just need a simple vanilla syslogd. Also there is no need to log so many files. Just dump them into /var/log/(cron/mail/messages)

remove rsystlog

Shell

```
apt-get -y remove --purge rsyslog
```

install syslogd

Shell

```
apt-get -y install inetutils-syslogd
```

Create a vanilla syslogd setup
Stop syslogd

Shell

```
service inetutils-syslogd stop
```

Remove old logs

Shell

```
for file in /var/log/*.log /var/log/mail.* /var/log/debug /var/log/syslog; do [ -f "$file" ] && rm -f "$file"; done
for dir in fsck news; do [ -d "/var/log/$dir" ] && rm -rf "/var/log/$dir"; done
```

Create syslog.conf

Shell

```
echo -e "*.*;mail.none;cron.none\t -/var/log/messages\ncron.*\t -/var/log/cron\nmail.*\t -/var/log/mail" > /etc/syslog.conf
```

Configure logrotate

Shell

```
mkdir -p /etc/logrotate.d
echo -e "/var/log/cron\n/var/log/mail\n/var/log/messages {\n\trotate 4\n\tweekly\n\tmissingok\n\tnotifempty\n\tcompress\n\tsharedscripts\n\tpostrotate\n\t/etc/init.d/inetutils-syslogd reload >/dev/null\n\tendscript\n}" > /etc/logrotate.d/inetutils-syslogd
```

Start syslogd

Shell

```
service inetutils-syslogd start
```

[/social-locker]

Set a static IP and remove DHCP-Client

Reduces CPU and memory.

Create a backup config

Shell

```
cp -f /etc/network/interfaces /etc/network/interfaces.dhcp-backup
```

Edit file: /etc/network/interfaces

replace

> *iface eth0 inet dhcp*

with

```
iface eth0 inet static
#set your static IP below
address 192.168.1.107

#set your default gateway IP here
gateway 192.168.1.1

netmask 255.255.255.0
network 192.168.1.0
broadcast 192.168.1.255
```

Reboot

Shell

```
shutdown -r now
```